**PROGETTO OOP a.a 2017-2018**

Gruppo Progetto : Di Egidio Laura 248831

Gentile Fabio 248809

Funzionalità del sistema:

Il nostro sistema deve gestire ed organizzare un insieme di collezioni, servizi e persone. L’ obiettivo del nostro sistema è la consultazione di manoscritti che devono essere digitalizzati. Le opere devono poter essere pubblicate e successivamente visionate da tutti gli utenti iscritti.

**1.1 Analisi dei requisiti**

Successivamente i requisiti per ogni sottosistema che siamo riusciti a trovare sono i seguenti:

**1 VIEWER**

* 1. **Login/registrazione** : Funzione per accedere al sistema da parte di utenti registrati. Possibilità di registrazione per nuovi utenti .
  2. **Ricerca** : Un utente può svolgere la ricerca nel catalogo per metadati (anno, titolo..) oppure all’ interno del testo della trascrizione. Funzione di ricerca dei testi attraverso criteri di scelta.
  3. **Download dell’ opera** : Operazione a disposizione solo per utenti premium (immaginiamo ad esempio che l’ utente premium è per noi un utente abbonato). Funzionalità che permette agli utenti premium di poter scaricare l’ opera.
  4. **ModuloTrascrittore** : Un utente può diventare un collaboratore e trascrivere testi compilando un modulo. Funzionalità per tutti gli utenti registrati di diventare trascrittori delle opere.
  5. **Dati Utente** : Funzionalità da parte dell’utente di poter accedere al proprio profilo e visualizzare i dati personali.

**2 UPLOADER**

**2.1 Gestione immagine** : In questo package distinguiamo il **Revisore Immagine** con le seguenti funzionalità:

* **Elimina immagine:** Eliminazione delle immagini non conformi alla qualità standard
* **Convalida immagine:** Funzione di convalida dell’ immagine scansionata dall’ Acquisitore.
* **Revisiona immagine:** Funzione di controllo dell’ immagine scansionata.

Distinguiamo ora l’ **Acquisitore** :

* **Scansione immagine** : Funzionalità che permette l’ acquisizione di immagini ad alta risoluzione dei manoscritti. L’ acquisitore caricherà le immagini in formato ‘jpg’ assegnandole all’opera di riferimento.

1. **TRANSCRIBER**
   1. **Gestione Testo** : In questo package distinguiamo il **Revisore Testo** e il **Transcriber:**

**Revisore Testo** :

* **Revisiona Testo :** Funzione da parte del Revisore di controllo del testo in formato TEI
* **Convalida Testo :** Funzione che permette al Revisore di convalidare il Testo trascritto
* **Modifica Testo :** Funzionalità che permette la modifica del testo da parte del revisore.

**Transcriber :**

* **Trascrive Testo :** Funzione di trascrizione testo da parte di trascrittori

**4 MANAGER**

4.1 **Assegnazione opere** : Tale funzionalità gestisce le assegnazioni di una o più opere ad uno o più trascrittori.

4.2 **Revisiona Trascrizioni** : Funzione di revisione delle trascrizioni da parte del Manager

4.3 **Pubblicazione :** Funzionalità di pubblicazione delle trascrizioni e delle opere.(IMMAGINI)

4.4 **Correzione Trascrizione** : Funzione di modifica delle trascrizioni da parte del Manager

4.5 **Convalida Testo** : Funzione di convalida delle trascrizioni da parte del Manager

4.6 **Assegna feedback :** (1-5) Funzione di gestione del livello dei trascrittori. Il livello più alto è per noi il punteggio 5, di base tutti i trascrittori partiranno con un livello standard di 3.

* 1. **Revisione immagine digitale** : Funzionalità di controllo dell’ acquisizione delle immagini

1. **ADMINISTRATOR**

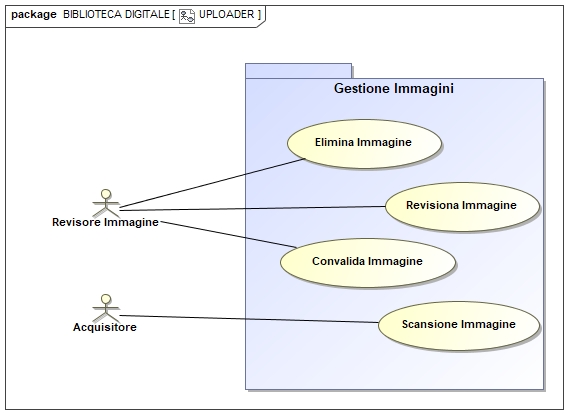
5.1 **Cancellazione Utente** : Funzionalità che permette di cancellare utenti dal sistema.

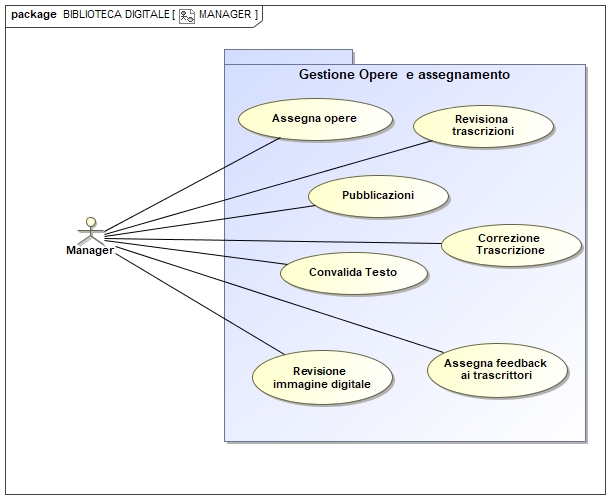
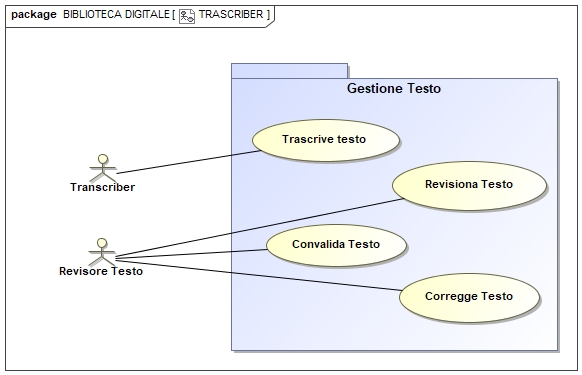
5.2 **Promozione Utente** : Funzione che permette di promuovere un Utente da Utente base a Trascittore oppure a Utente Premium

5.3 **Inserimento Utente** : Funzione che permette l’ inserimento di utenti all’ interno del sistema.

1.2 MODELLI UML USE CASE

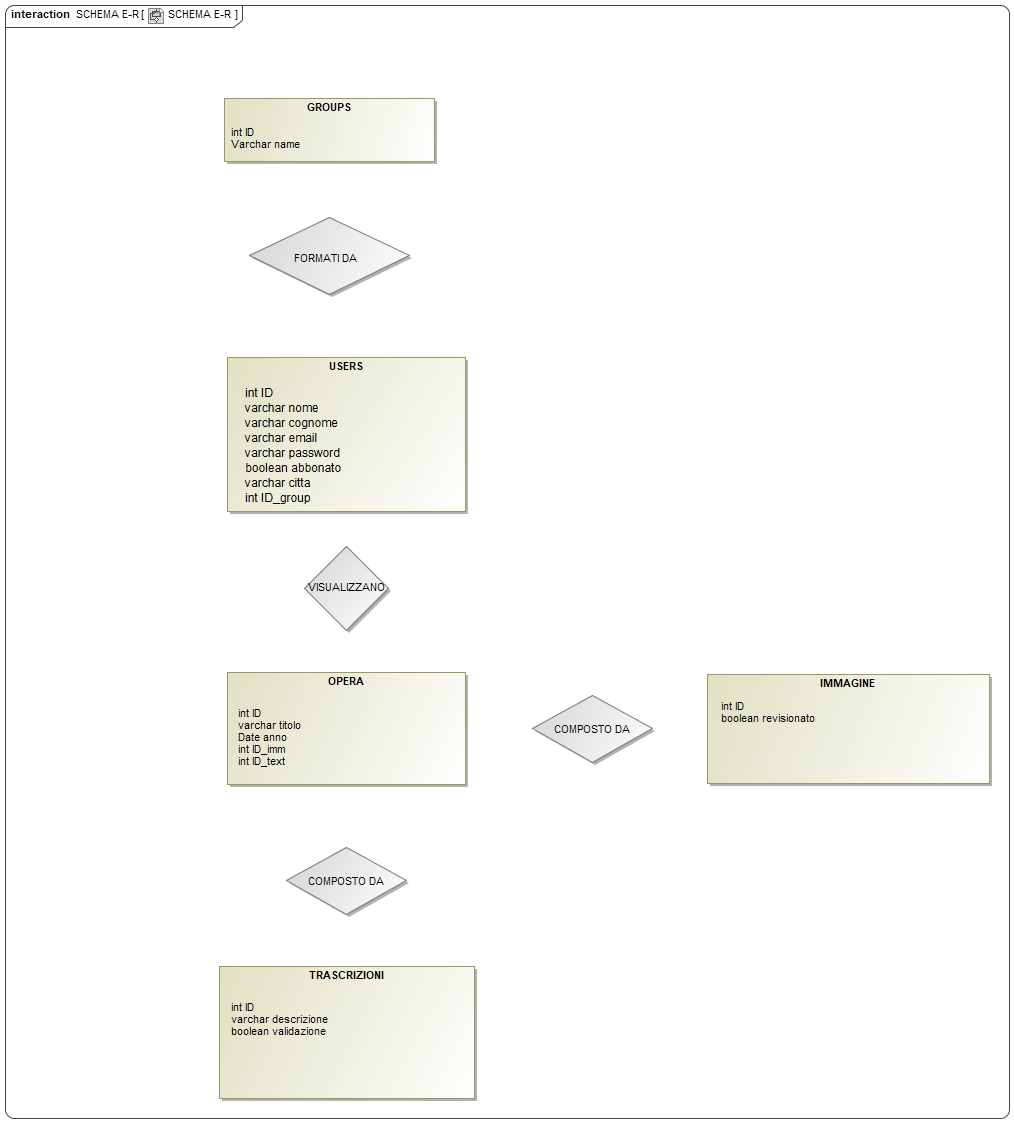
Per definire al meglio lo Use Cases Diagram completo abbiamo di nuovo suddiviso gli attori con le loro funzionalità nei sottosistemi





UML USE CASES COMPLETO

Prima di determinare il Modello di Dominio è stato utile, per una maggior visione delle entity, crearci lo schema ER.

****

(1,n)

(1,n)

(1,1)

(1,1)

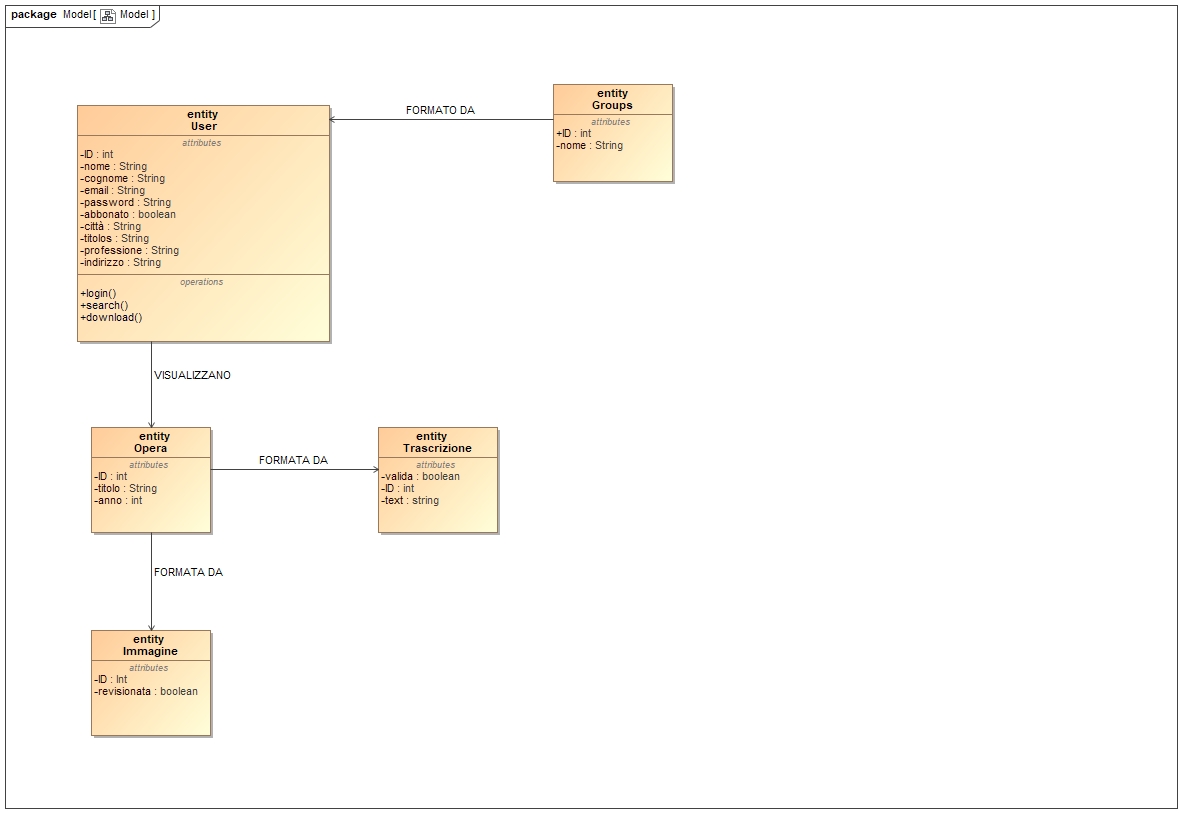
(1,n)

(1,n)

(1,1)

(1,n)

1.3 DOMAIN MODEL



1.4 *analisi finalizzata all’individuazione di classi entity, boundary*

*e controller*

ENTITY

|  |  |  |
| --- | --- | --- |
| ENTITY | ATTRIBUTI | DEFINIZIONE |
| Groups | ID  Nome | In base all’ ID e al nome del tipo ‘manager, trascrittore...’ capiamo quale ruolo svolgo il nostro utente all’interno del sistema. Tutti gli utenti sono per noi utenti registrati ma che in base al ruolo definito appunto da Groups siamo in grado di attribuirgli determinate azioni. |
| Users | ID  Nome  Cognome  Abbonato  Email  Password  Citta  Indirizzo  Titolo di studio | Nella classe Users l’utente può visualizzare il proprio profilo determinato dai seguenti attributi. Inoltre, l’utente può svolgere le azioni quali: login,  registrazione, modifica,  revisione, pubblicazione,  download, trascrizione,  acquisizione, feedback… |
| Opera | ID  Titolo  Anno | Opera formata da un codice ID, il titolo dell’opera e l’anno di pubblicazione. |
| Immagine | ID  Revisionato | Nell’immagine abbiamo l’attributo revisionato (boolean) da dove specifichiamo se la scansione di una certa opera è stata revisionata successivamente |
| Trascrizione | ID  Convalida  Text | Anche la trascrizione ha un attributo booleano ‘convalida’ che specifica se la trascrizione sia stata o meno convalidata. |

|  |  |
| --- | --- |
| CheckImage() | Utilizzato dal revisore per la convalida dell’immagine acquisita |
| ScanImage() | Scansione immagine da parte dell’ acquisitore |
| CheckText() | Validazione del testo digitale da parte del revisore testo |
| TrascrText() | Utilizzato per la trascrizione dal trascrittore |
| PubOp() | Pubblicazione dell’ opera da parte del manager e dal revisore dell’ immagine in caso non ci siano trascrizioni |
| Select\_Op() | Il manager decide di assegnare un opera al trascrittore |
| Modulo\_Online() | L’utente compilando un modulo online diventa un trascrittore delle opere |
| Revisiona() | Il manager revisiona l’ immagine digitale e il testo trascritto |
| ButtonLogin() | Pulsate per accedere al profilo dell ‘ Utente |
| ButtonDownload() | Pulsante per effetture il download dell’opera |

BOUNDARY

CONTROL OBJECT

Analizzando le funzionalità del sistema abbiamo individuato diversi controlli che il sistema dovrà fare all’ utente in diverse occasioni.   
ManagerControl() : Controllo utilizzato nel momento in cui un utente voglia modificare opere, assegnare trascrizioni, dare feedback ai trascrittori o pubblicare un’ opera.  
PremiumUsersControl() : Controllo per gli utenti che vogliono scaricare l’ opera. Visualizza se l’utente è un utente abbonato (Premium) o viceversa.  
UsersAcquisizioneControl() : Controlla se un utente può o meno acquisire un immagine.  
UsersTrascrControl(): Controlla se un utente può o meno trascrivere il testo dell’ opera.  
RevisoreControl(): Controlla se l’utente è un revisore. In questo caso verifica se è un revisore Trascrittore o revisore Acquisizione. Da ciò può dunque revisionare e convalidare l’immagine/testo.  
UsersRegisteredControl() : Controlla nel momento del login se l’utente è registrato.

2 SYSTEM DESIGN

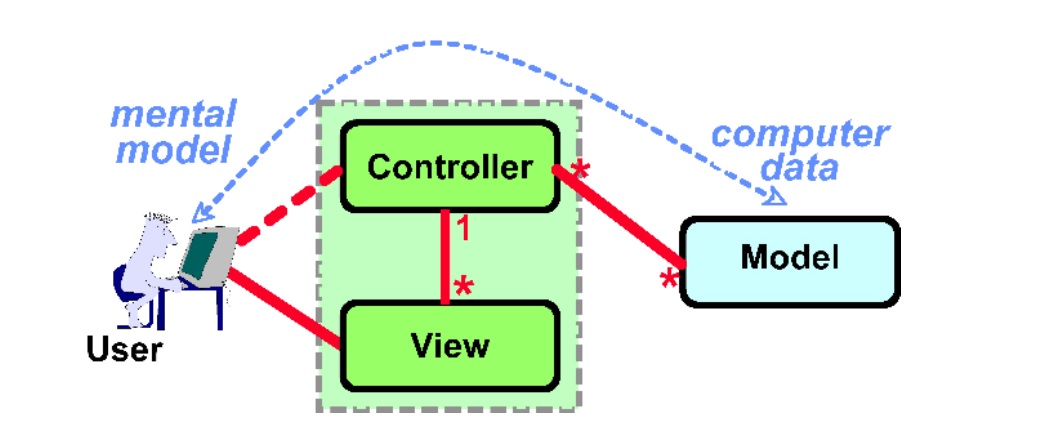
ARCHITETTURA SOFTWARE

È stato scelto dal nostro gruppo il pattern MVC (Model View Controller), in quanto permette una

suddivisione del sistema in parti indipendenti tra di loro più facili da gestire.

Il pattern architetturale è così diviso

* Rappresentazione del modello di dominio (Model)
* Interfaccia Utente (View)
* Controllo operazioni Utente-Sistema (Controller)



MVC è un pattern architetturale che ci permette di separare la logica di presentazione dei dati da quella di business.

I componenti software sono divisi in base ai compiti svolti all’interno del sistema

* Il Model fornisce i metodi per accedere ai dati utili. Gestisce direttamente i dati, la logica e le regole dell’applicazione.
* Il View si occupa della visualizzazione dei contenuti. È una qualsiasi rappresentazioni in output di informazioni, nel nostro esempio le immagini, il testo ecc..
* Il Controller riceve i comandi dall’ utente attraverso il view e li svolge alterando lo stato degli altri due componenti.

